(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
16 August 2001 (16.08.2001)

PCT

(10) International Publication Number
WO 01/59586 A2

(51) International Patent Classification[7]: G06F 17/00

(21) International Application Number: PCT/CA01/00022

(22) International Filing Date: 9 January 2001 (09.01.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/500,596 10 February 2000 (10.02.2000) US

(71) Applicant *(for all designated States except US)*: **DWL INCORPORATED** [CA/CA]; 230 Richmond Street East, Level 2, Toronto, Ontario M5A 1P4 (CA).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **VAN RUN, Paul, A., R.** [CA/CA]; 26 Martin Crescent, Toronto, Ontario M4S 2V4 (CA). **LADHA, Karim, A.** [CA/CA]; 5 Field Sparroway, Unit 305, Willowdale, Ontario M2H 3B6 (CA).

(74) Agent: **GRAY, Brian, W.**; Blake, Cassels & Graydon LLP, Box 25, Commerce Court West, Toronto, Ontario M5L 1A9 (CA).

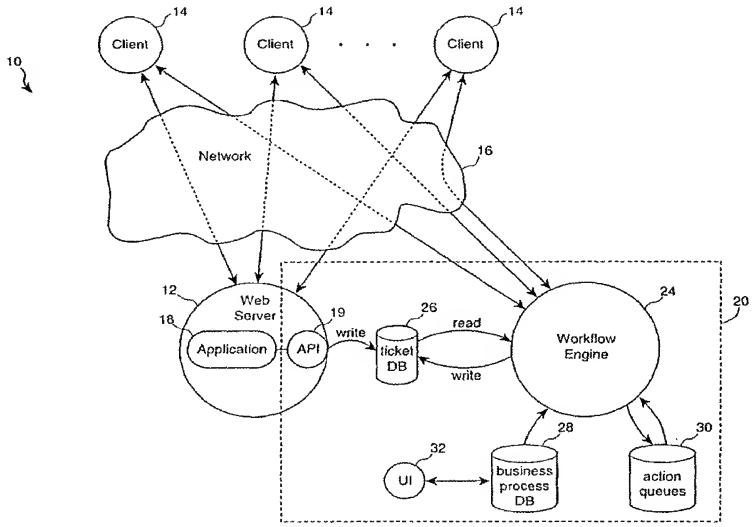
(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:
— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: WORK-FLOW SYSTEM FOR WEB-BASED APPLICATIONS



(57) Abstract: A work-flow system for supporting a web-based application in which a server runs an application program accessible by a plurality of client computers via a network. The work-flow system comprises a database for specifying one or more business processes and at least one action associated with each business process. Each business process is initiated via a ticket stored in a ticket repository of the database. An interface is provided for enabling the application program to write tickets to the ticket repository. The system further includes a work-flow engine for matching the tickets with business processes, and in the event of a match, for executing the corresponding actions. In this manner, "back office" processing tasks can be handled asynchronous of the application program, thereby reducing the processing load on the application program and web server and enhancing the responsiveness of the application to its users.

WO 01/59586 A2

WORK-FLOW SYSTEM FOR WEB-BASED APPLICATIONS

FIELD OF INVENTION

The invention relates to the field of work-flow systems, and more specifically to work-flow systems for facilitating electronic commerce or network centric applications.

BACKGROUND OF INVENTION

As a result of the rapid growth of the Internet many companies have built or are in the process of building web sites in order to more efficiently interact with their customers and clients. Many organizations are also seeking to leverage the benefits of e-commerce applications within their own organizations, and have built a variety of internal applications based on the Internet paradigm. In association with such frontline applications, there may be a variety of 'back office' processing functions which may require the collaboration of many individuals to carry out, and for this purpose it is often desired to use some sort of workflow system to manage and facilitate this process.

A variety of work-flow systems are known in the art. In a traditional work-flow system, some initial document, e.g., an insurance policy application form, is digitized and then passed around to a variety of people involved in processing the document in conjunction with the internal practices of the organization in question. These traditional work-flow systems, however, are typically quite bulky and not particularly well suited to supporting online business applications and facilitating seamless e-commerce transactions. Accordingly, a work-flow system geared towards facilitating web-based applications is desired.

SUMMARY OF INVENTION

Broadly speaking, the invention relates to a work-flow system from supporting a web-based application in which a server runs an application program accessible by a plurality of client computers via a network. The work-flow system comprises a database for specifying one or more business processes and at least one action associated with each business process. Each business process is initiated via a ticket stored in a ticket repository of the database. An interface is provided for enabling the application program to write tickets to the ticket repository. The system further includes a work-flow engine for matching the tickets with business processes, and in the event of a match, for executing the the corresponding actions. In this manner, 'back office' processing tasks can be handled asynchronous of the application program, thereby reducing the processing load on the application program and web server. Since the application program provides the virtual point of contact with the customer or client, it thus possible to also increase the responsiveness of the application to them.

Among the actions provided by the work-flow system is a URL access. This feature can be employed in such as way as to mimic a user filling out an electronic form available over the network. This fosters the re-use of the application program and facilitates business to business e-commerce as exemplified in greater detail below. Other actions supported by the system include sending electronic mail to a destination on the network, transferring a file to a destination on the network, creating new tickets, and updating an external database with information passed to the system by the application program.

In the illustrative embodiment described herein, the database includes an action definition repository which specifies a source for the data employed in the actions, and the work-flow engine parses the action definition repository in order to create specific actions such as those outlined above. This feature increases the versatility of the work-flow system.

BRIEF DESCRIPTION OF DRAWINGS

The foregoing and other aspects of the invention will become more apparent from the following description of specific embodiments thereof and the accompanying drawings which illustrate, by way of example only, the principles of the invention. In the drawings, where like elements feature like reference numerals:

Fig. 1 is an architectural block diagram of the software components of a network-based application and work-flow processing system;

Fig. 2 is a schematic diagram exemplifying the relationship between a web-based application and business processes executed by the work-flow system;

Fig. 3 (comprising Figs. 3a - 3d joined in the indicated manner) is a diagram of a database schema employed by a preferred embodiment of the work-flow system; and

Fig. 4 is a flowchart showing the processing activities carried out by the work-flow system.

DETAILED DESCRIPTION OF ILLUSTRATIVE EMBODIMENT

Fig. 1 shows the software architecture of a network-based application and work-flow processing system 10. The system 10 comprises a central computer or web server 12, which communicates with a plurality of client computers 14 through a communications network 16. The web server 12 is associated with an address on the network 16, which may be the public Internet or an organizational specific intranet. The web server 12 and client computers 14 communicate using one of the popular versions of hyper text mark-up languages, e.g., HTML, and to this end the client computers run browsers such as Netscape™ (TM - Netscape Communications Corporation) or Internet Explorer™ (TM - Microsoft Corporation). The invention is not, however, limited to these communication protocols.

The web server 12 hosts a web-based application 18, the functionality of which will be specific to each organization and what it is attempting to accomplish. For example, a bank or insurance company may have an application which enables customers to apply for a new account or policy using online electronic or "web-based" forms. In another example of an application, retailers or wholesalers may provide an online product catalogue to enable customers to purchase products online for future delivery to them. In these illustrative examples, the customer activities will spawn a wide variety of subsequent "back office" processing tasks within the organization which will likely require collaboration amongst a variety of individuals. For instance, when a new bank account or insurance policy is applied for, the company may require that:

(a) a message be sent to the customer advising him or her that the application has been received;

(b) an on-line credit check be carried out;

(c) new account information is entered into the company's legacy computer systems so that an account number can be issued; and

(d) a clerk is alerted about the new account in order to send a check book or policy to the customer.

Similarly, fulfilling the purchase of an online product may require the collaboration of a number of persons, including a warehouse clerk who attends to retrieving the product from a warehouse and a shipping clerk who attends to filling out the paperwork for a courier company. Internally, the merchandiser may also have to attend a number of maintenance operations associated with their business, such as the continuous maintenance of the catalogue, including adding, editing or deleting products or pages of the catalogue.

The invention enables many of the non-critical back-office processing steps to be carried out asynchronously of the web-based application 18, and also in a way that does not require that the electronic document or other data accumulated by the application 18 to be electronically moved or copied to the individuals associated with the back office

processing. For instance, in the account-opening procedure described above, the initial message sent to the client may include an electronic document sent to the customer via ftp file transfer protocol. This make take some time, and such a non-critical task may be handled asynchronous of the web-based application 18. Since the application 18 functions as the virtual 'point of contact' with the customer or user, removing this and similar tasks from the application 18 will reduce the computational load thereof and potentially enhance the responsive of the system as perceived by its users.

For these purposes, the system 10 includes a work-flow subsystem 20 comprising a work-flow engine 24, a ticket repository or database 26, a business process repository 28, and action tables or queues 30. The business process repository 28, stores the logic or rules associated with various business processes such as the back office organizational functions described above. A user interface 32 enables business analysts to establish these rules and populate the business process repository 28. The ticket repository 26 stores "tickets" which initiate business processes. Tickets are created by the web-based application 18 in order to initiate specific business processes, and a ticketing application program interface (API) 19 is provided within the web server 12 for this purpose. The workflow engine 24 processes the business rules and tickets stored in repositories 26, 28 in order to create action items, which are stored in queues 30. Actions which are supported in the illustrative embodiment include: e-mail messages, notifications, file transfers, web page accesses, database updates, and the creation of more tickets, all as described in greater detail below. The workflow engine 24 is connected to the network 16 in order to support some of these actions.

Fig. 2 exemplifies the relationship between a web-based application, business processes, and actions. In this example, the application is electronic retailing. A certain portion of the application enables company personnel to maintain an online catalogue via an HTML form 40. The form 40 provides input/display fields 42, 44, and 46 in relation to a

textual description of the product, a photo-graphical image of the product, and the cost to the consumer, respectively. In additions, input/display fields 48, 50 and 52 relate to the identity of the supplier of the product, the supplier's product code, and the supplier's selling price, respectively. At the bottom of the form 40 virtual buttons 54, 56 and 58 are provided for creating a new catalogue entry, editing an existing catalogue entry, and deleting the displayed catalogue entry, respectively. (This example presumes that a previous form enabled the user to search for a particular product or add a new one, following which form 42 is displayed for use with all three functions.) The source code within application 18 for form 40 is schematically shown at 60. A portion 62 of the code is responsible for displaying the form 40 and a portion 64 of the code is responsible for processing the form once the input/display fields 42 – 52 are populated and one of the virtual buttons 54 - 58 is actuated. Within portion 64 the ticket API 19 is invoked three times (at 64a, 64b and 64c) in order to create tickets which initiate the following business processes: Maintenance/CreatePage 66; Maintenance/EditPage 68; and Maintenance/DeletePage 70. These business processes are carried out by the work-flow subsystem 20 and are shown separately. In the illustrated embodiment, the business process repository 28 has been defined such that the Maintenance/CreatePage 66 comprises the following four actions 72 – 78:

(a) management notification, so that an authority can review the contents of the new page and suggest corrections if required;

(b) messaging foreign offices, so that a translator in one or more foreign offices can create a corresponding new entry in a foreign language catalogue;

(c) updating a legacy inventory database; and

(d) ordering a predetermined quantity from the supplier.

In action item (a), the authority is given one (1) week in which to review the contents of the new page before another reminder is sent. The other business processes 68 and 70 may require similar actions, although these are not explicitly shown.

Referring additionally to Figs. 3 and 4 wherein a database schema of repositories 26, 28 and 30 and a flowchart of the work-flow engine 24 are shown, the manner

in which the illustrative work-flow subsystem 20 operates is explained in conjunction with the example set out in Fig. 2.

As shown in Fig. 3, the ticket repository 26 comprises a WFTickets table 80 which is linked via an "id" field 80a to a WFTicketParameters table 82 and a WFReferences table 84. (Linked fields are shown in the database schema with the legend "(FK)", which signifies a foreign key). The records of tables 80, 82 and 84 collectively represent a ticket. Each ticket is uniquely identified by the "id" field 80a, which functions as the primary key of table 80. A new key value is created every time the web based application 18 issues a ticket.

The WFTickets table 80, provides the basic parameters of each ticket and includes, *inter alia*, an "origin_business_process" field for identifying a corresponding business process; an "action_date" field for specifying the date and time the corresponding business process is scheduled to begin; and a "status" field for indicating whether the ticket has been processed or not. Tables 82 and 84 enable additional information, which may vary from ticket to ticket, to be passed from the application 18 to the work-flow sub-system 20. For instance, the WFReferences table 84 includes a "type" field for specifying one of a predetermined type of variable passed, e.g., URL address or product code, as well as a "reference" field which contains the variable information. The primary key 84a of table 84 is linked to a "reference_id" field in table 80. Similarly, the WFTicketParameters table 82 includes a "name" field and a "value" field for enabling the name of a parameter and its value to be passed. Thus, for instance, a ticket may be created with WFReferences.type := "product code", WFReferences.reference := "book", and WFTicketParameters.name := "category" and WFTicketParameters.value := "science fiction".

The business process repository 26 comprises a WFBusinessProcess table 86, a WFProcessDefinition table 88, and a plurality of action definition tables 90, 92, 94, 96, 98,

and 100, all of which are interlinked. Generally speaking, a business process and its related actions, such as process 66 and actions 72 – 78 of Fig. 2, are defined by configuring tables 86 – 100.

The WFBusinessProcess table 86 comprises fields which enable business analysts to specify distinct business processes via a unique primary key "id" field, which is linked to the "origin_business_process" field of the WFTickets table 80. The remaining fields of table 86 are for descriptional purposes only so as to enable business analysts to identify a business process by something other than a mere number. For instance, the analyst may assign WFBusinessProcess.application:= "catalogue", WFBusinessProcess.process:= "maintenance", and WFBusinessProcess.operation:= "create page" to correspond to the Maintenance/CreatePage business process 66 shown in Fig. 2.

The WFProcessDefinition table 88 specifies exactly which tickets will trigger or initiate a given business process and includes the following fields:

- "id" – a unique identifier
- "on_operation" – linked to WFBusinessProcess.id and WFTickets.origin_business_process
- "on_reftype" – linked to WFReferences.type
- "on_user" – linked to WFTickets.user
- "on_status" – linked to WFTickets.status
- "on_time" – linked to WFTickets.action_date
- "status" – specifies whether the business process is active or not
- "valid_date_from" – specifies a starting date for a period within which the business process is applicable
- "valid_date_to" – specifies a termination date for the period within which the business process is applicable.

Each of the action definition tables 90-100 includes a "processdef_id" field linking the table to the WFProcessDefinitions table 88. An m:1 relationship is permitted between each action definition and a business process definition. Generally speaking, the action definition tables specify the data sources for corresponding actions, thereby providing a level of data indirection. This is made possible through the use of a small interpretative language employed by the work-flow subsystem 20 in the course of evaluating the "source_..." fields of the action definition tables 90 – 100. For example, the WFDefURL table 94 includes a "source_URL" field into which an expression can be entered that will be evaluated or parsed at runtime to indicate the network address of a particular page on the Internet or intranet. The language provides predetermined tokens for specifying the location of data, including:

(e) a literal token which specifies that the data for the source field follows the token;

(f) a token indicating that the data should be obtained from a specified field of the corresponding ticket (i.e., any field of tables 80, 82 and 84); and

(g) a token which specifies that the data should be obtained from a field in an external database.

Thus, for example, in furtherance of action 78 (Fig. 2), the URL of a supplier from which the new product may be ordered from may be determined by having the application 18 pass the name of the supplier to the WFTicket Parameters table 82 and thereafter consulting an external database comprising the names and URL addresses of all of the organization's approved suppliers. In this example, field WFDefURL.source_URL may be set to: <%DB(Supplier, <%supplier-name%>, Supplier.address), where %DB(arg1, arg2, arg3) is a token indicating a lookup in an external database arg1, based on key arg2, returning a value arg3. The <%supplier-name%> indicates a lookup in the WFTicket

Parameters table 82 for a record in which the "ticket-id" field matches the id of the corresponding ticket and the "name" field matches "supplier-name", in which case the data in the "value" field of that record is returned.

Once the action definition tables 90 – 100 are parsed, the corresponding action items are stored in queues 110 – 120. (Note that the structure of each discrete queued item is shown in Fig. 3.) The work-flow engine 24 will then execute the corresponding actions in accordance with the value of the "action_date" fields, which specify the date and time the action item is scheduled for execution.

The foregoing may be better appreciated by referring to the flow chart of Fig. 4. The work-flow engine 24 features a main execution thread which, in an initial step 150, selects a record of the WFProcessDefinitions table 88 for processing. Each such record represents a discrete business process, as discussed above. The records of table 88 may be processed sequentially in round-robin fashion or alternatively based on a "priority" field therein. Irrespective of how the record/business process is chosen, the work-flow engine 24 executes a query in the WFTickets table 80 based on the fields of table 88 in order to find all matching tickets in table 80. The query is built on the fly based on the contents of the current record in table 88. An illustrative SQL query statement is:

```
SELECT * FROM WFTickets
        WHERE origin-business_process = WFProcessDefinition.on_operation
        AND status = WFProcessDefinition.on_status
```

At step 154 the result of the query is tested to see if corresponding tickets exist for the record/business process under consideration. If not, control returns to step 150. If one or more corresponding tickets exist, the corresponding business process is initiated. Step 154 then queries each of the definition action tables 90 – 100 to determine all actions

associated with the initiated business process. An illustrative SQL query statement for the URL action definition (i.e., WFDefURL table 94) is:

```
SELECT * FROM WFDefURL
    WHERE processdef_id = WFProcessDefinitions.id
    ORDER BY order-number.
```

Similar queries are conducted on the other action definition tables.

Once all the corresponding action definition records are found for the business process under consideration, the "source" fields thereof are parsed at step 156 and new action items (i.e., records) are added to the action tables or queues 110 – 118. The work-flow engine 24 comprises additional threads of execution for processing these queues. These include an SQL processor 160, a message processor 162, an FTP processor 164 and a URL processor 166. Each of the processor periodically scans its corresponding queue to determine when the action items stored therein are scheduled for execution based on an "action_date" field in each of tables 110, 112, 114 and 116. Once an item is ready for execution, the processors will then execute an SQL statement, send an e-mail, transmit a file, and initiate a web-page access, as appropriate.

One exception to this general scheme relates to the action of creating a new ticket. This is executed immediately by the work-flow engine 24 and the resulting ticket is stored in WFTickets table 80 for later processing by the main thread of the engine 24. The creation of new tickets enables one business process to chain or link another business process. For instance, in the example of Fig. 2 a first ticket (as subsequently exemplified) is issued by the application program in respect of business process 66 and action 72 - management review of the newly created catalogue page. A second business process may be defined in which, as a general rule, management has one week to review a business decision.

This business process may be initiated by having business process create a second ticket linked to the second business process.

Another exception to the general scheme relates to the WFNotifications table 118, which employs a more passive form of execution. In the illustrative embodiment, a notification is a textual or multi-media message, which is stored in table 118 until such time as its intended recipient decides to retrieve the message. To facilitate this, the API 19 provides application 18 with a function which enables the application to display an icon, such as a light, which blinks when a message exists for the user (whether as a single recipient or as part of a larger group). Preferably, separate icons are displayed for each waiting message, and a brief title associated with the message may also be displayed. When the icon is pressed, the WFNotifications table 118 is searched to find the corresponding message(s) applicable for that individual.

Having now generally described the database scheme and the operation of the work-flow engine 24, an example is presented as to how the work-flow subsystem 20 may be set up to accomplish business process 66 and its related actions shown in Fig. 2.

First, a new record in the WFBusinessProcess table 86 may be set up as shown below:

- WFBusinessProcess.application := "Catalogue"
- WFBusinessProcess.process := "Maintenance"
- WFBusinessProcess.operation := "Create Page"
- WFBusinessProcess.enabled := "y"

The WFTicket Parameters table 82 can be used to receive the URL of the new online catalogue page from the application 18, as follows:

- WFTicketParameters.name:= "new-page"
- WFTicketParameters.value:= (URL address passed by application 18)

Assuming that the system sets WFBusinessProcess.id for the above record to some number 123, a record in the WFProcessDefinition table can be set up as follows:

- WFProcessDefinitions.on_reftype := null
- WFProcessDefinitions.on_user := null
- WFProcessDefinitions.on_status := null
- WFProcessDefinitions.on_time := null
- WFProcessDefinitions.status := valid
- WFProcessDefinitions.priority := null
- WFProcessDefinitions.on_operation := 123

To accomplish action 72, the notification to management, a record in the WFDefNotifications table 100 can be set up as follows (assuming that the system sets WFProcessDefinitions.id to 456):

- WFDefNotifications.processdef_id: = 456
- WFDefNotifications.order_number : = 0 (this field establishes the order of processing when more than one record is created in the table as a result of a single business process.)
- WFDefNotifications.source_title: = "review"
- source_user_id: = "Manager"

- source_origin_user: = <%WForigin_user%> (which token indicates a lookup in field "origin-user" in the corresponding ticket table 80)
- source_description := "Please review the new product added to the catalogue at" <%new_page%>

The application 18 can initiate this business process by creating a ticket with:

- WFTickets.origin_business_process := 123
- WFTickets.origin_user := (user name of person creating new catalogue page)
- WFTickets.action_date := (the date/time the ticket is created)
- WFTickets.status := "valid"
- WFTickets.priority := null

This will cause the work-flow engine 24 to find record 456 in the WFProcessDefinitions table 88, which in turn will cause the just-described record in the WFDefNotifications table 100 to be parsed. As a result, a notification message will be created from the user who created the new page to the manager, wherein the body of the message specifies the URL of the newly created page. When the manager reads the message, the manager's browser will enable him or her to click on the address of the newly created page and jump to it immediately.

Similarly, action 74, notifying foreign offices, can be accomplished by setting up a similar record the WFDefMessages table 96. If desired, a copy of the newly created page can transferred over to the foreign office by setting up a record in the WFDefFTP table 98. In this case WFDefFTP.source_subdirs can be set to <% file-name%> to specify the file

to be transferred, where <%file-name%> indicates a lookup in the WFTicketParameters table 82 for field "name" = "file-name".

Action 76 in respect of updating an external inventory database can be accomplished by having the application 18 pass the product code of the new product in the value field of the WFTicketParameters table 82. A record can then be set up in the WFDefDB table 92 with WFDefDB.source_SQL_command being set to an appropriate SQL command based on the new product code stored in the WFTicketParameters table.

Action 78 in respect of ordering a predetermined quantity of the new product can be carried out through appropriate configuration of the WFDefURL table 94. The present invention enables a reverse "screenscraping" technique to be carried out with respect to web pages. More specifically, when an HTML form is displayed, a browser typically returns the data that has been filled in by the user to the host server by reinitiating a call or jump to the URL address of the form, followed by a list of the fields and their values. The generally used format is URL + (field 1, value 1) + (field 2, value 2), ... Thus, it is possible to remotely and automatically fill in a web-based form through knowledge of its structure, which most popular browsers can relatively easily provide. In the illustrated embodiment, the WFDefURL table 94 is linked to a WFDefURLPost table 94B which is designed to store the variable field names and field values of a web-based form. Accordingly, in the present example, a predetermined quantity of the new product can be automatically ordered by the work-flow subsystem 20 by specifying the URL of the supplier in WFDefURL.Source_URL, either literally or by using a data indirection operation as discussed above. The field names and field values of the supplier's web-based order entry form are stored in the WFDefURLPost table 94B, again either literally or through the data indirection operation. Once parsed, the URL address will be stored in the WFActionURL queue 112, and the parsed field names and field values will be stored in a linked WFActionURLPost table 112B. When the URL processor 166 processes the URL queue 112, the processor 166 will search

for corresponding entries in table 112B, and build a URL access string to the supplier's web page in the form URL + (field 1, value 1) + (field 2, value 2) ..., wherein the field names and field values have been set to order a predetermined amount of the new product. In this manner, the work-flow subsystem 20 can mimic a user filling out an electronic form made available over the network, and thereby facilitate seamless business to business e-commerce transactions.

Those skilled in the art will appreciate that the example applications presented herein are for descriptional purposes only and are not intended to limit the invention to these examples. Similarly, it will be understood by those skilled in this art that numerous variations and modifications may be made to the embodiments described herein without departing from the spirit of the invention.

CLAIMS

1. A web-based work-flow system, comprising:

a database for specifying one or more business processes and at least one action associated with each business process, wherein each business process is initiated via a ticket stored in a ticket repository of the database;

an application program interface for writing tickets to the ticket repository; and

a work-flow engine for matching tickets with business processes, and in the event of a match, for executing the corresponding actions.

2. The system according to claim 1, wherein said actions include filling out an electronic form available over the network.

3. The system according to claim 1, wherein said actions includes sending electronic mail to a destination on the network and transferring a file to a destination on the network.

4. The system according to claim 1, wherein said actions include updating an external database with information passed to the system by an application program.

5. The system according to claim 1, wherein the database includes an action definition repository which specifies a source for the data employed in said actions, and wherein the work-flow engine parses the action definition repository in order to create said actions.

6. A processing system, comprising:

a server running a web-based application program;

a network;

a plurality of client computers connected to the server via the network;

a work-flow subsystem comprising a database for specifying one or more business processes and at least one action associated with each business process, wherein each business process is initiated via a ticket stored in a ticket repository of the database; an interface for enabling the application program to write tickets to the ticket repository; and a work-flow engine for matching tickets with business processes, and in the event of a match, for executing the corresponding actions asynchronous of the application program.

7. The system according to claim 6, wherein said actions include filling out an electronic form available over the network.

8. The system according to claim 6, wherein said actions includes sending electronic mail to a destination on the network and transferring a file to a destination on the network.

9. The system according to claim 6, wherein said actions include updating an external database with information passed to the system by an application program.

10. The system according to claim 6, wherein the database includes an action definition repository which specifies a source for the data employed in said actions, and wherein the work-flow engine parses the action definition repository in order to create said actions.

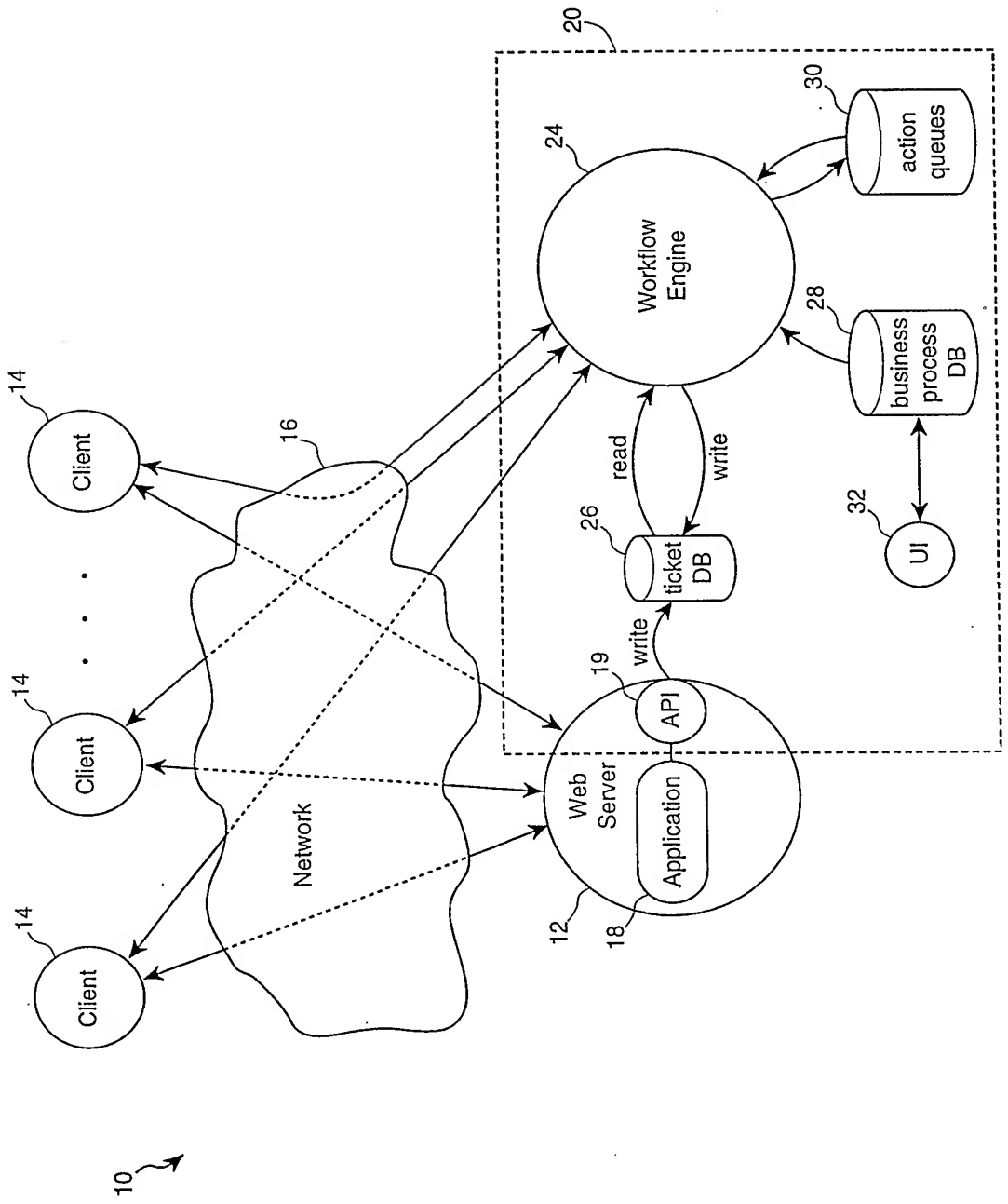

10
Client
Client
Client
14
16
Network
12
Web Server
Application
18
API
19
write
20
Workflow Engine
24
read
write
ticket DB
26
business process DB
28
action queues
30
UI
32

Figure 1

2/8

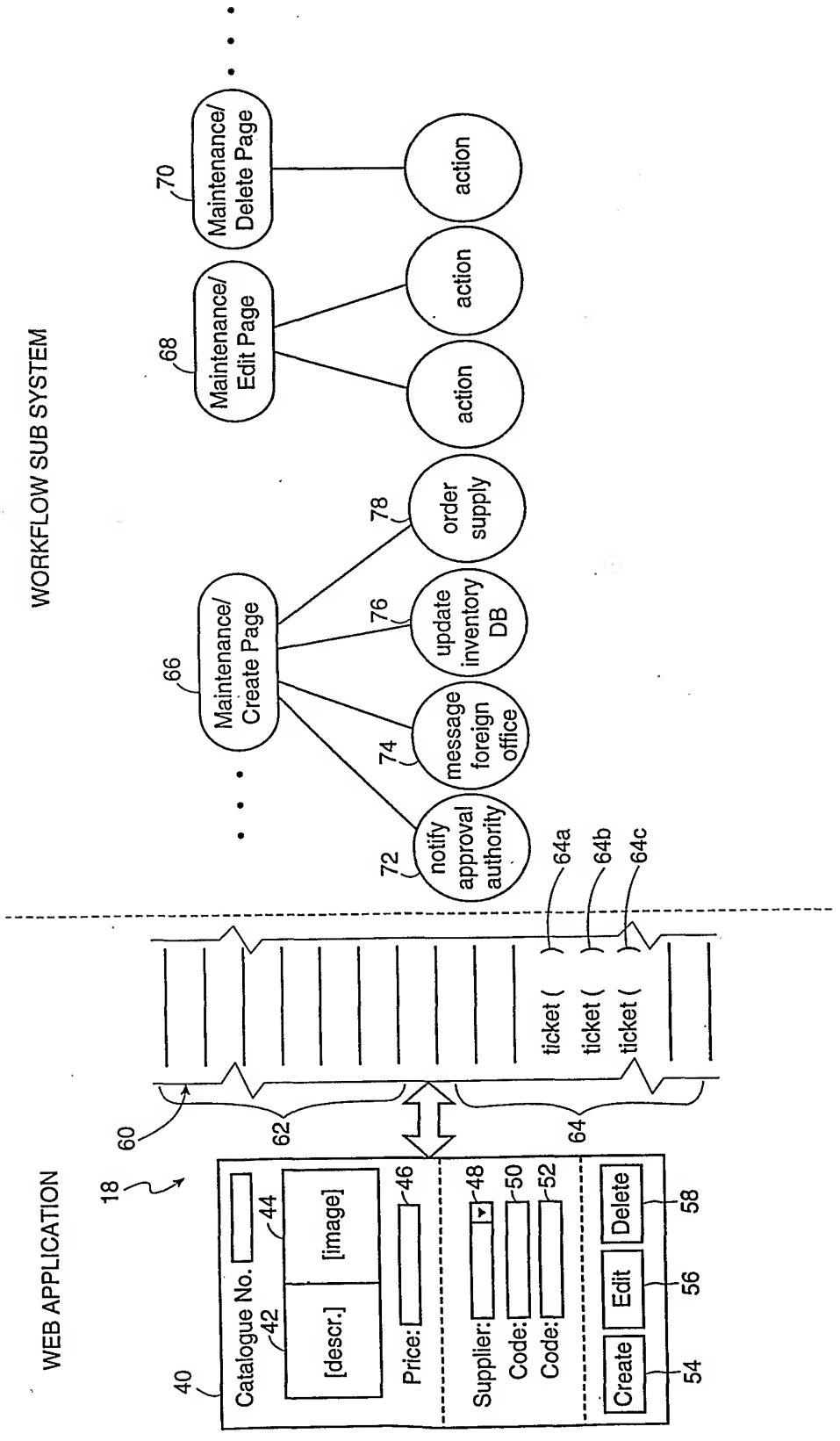

WEB APPLICATION
WORKFLOW SUB SYSTEM
18
40
Catalogue No.
42
[descr.]
44
[image]
Price:
46
Supplier:
48
Code:
50
Code:
52
Create
54
Edit
56
Delete
58
60
62
64
ticket (
64a
ticket (
64b
ticket (
64c
66
Maintenance/ Create Page
68
Maintenance/ Edit Page
70
Maintenance/ Delete Page
72
notify approval authority
74
message foreign office
76
update inventory DB
78
order supply
action
action
action

Figure 2

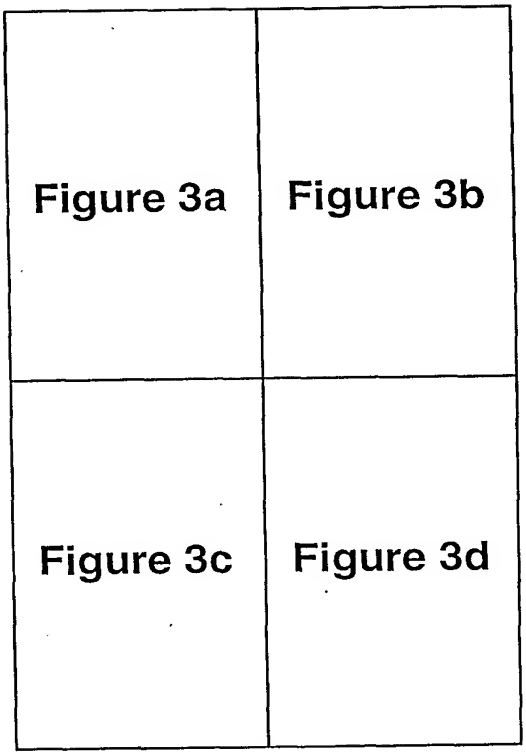


Figure 3

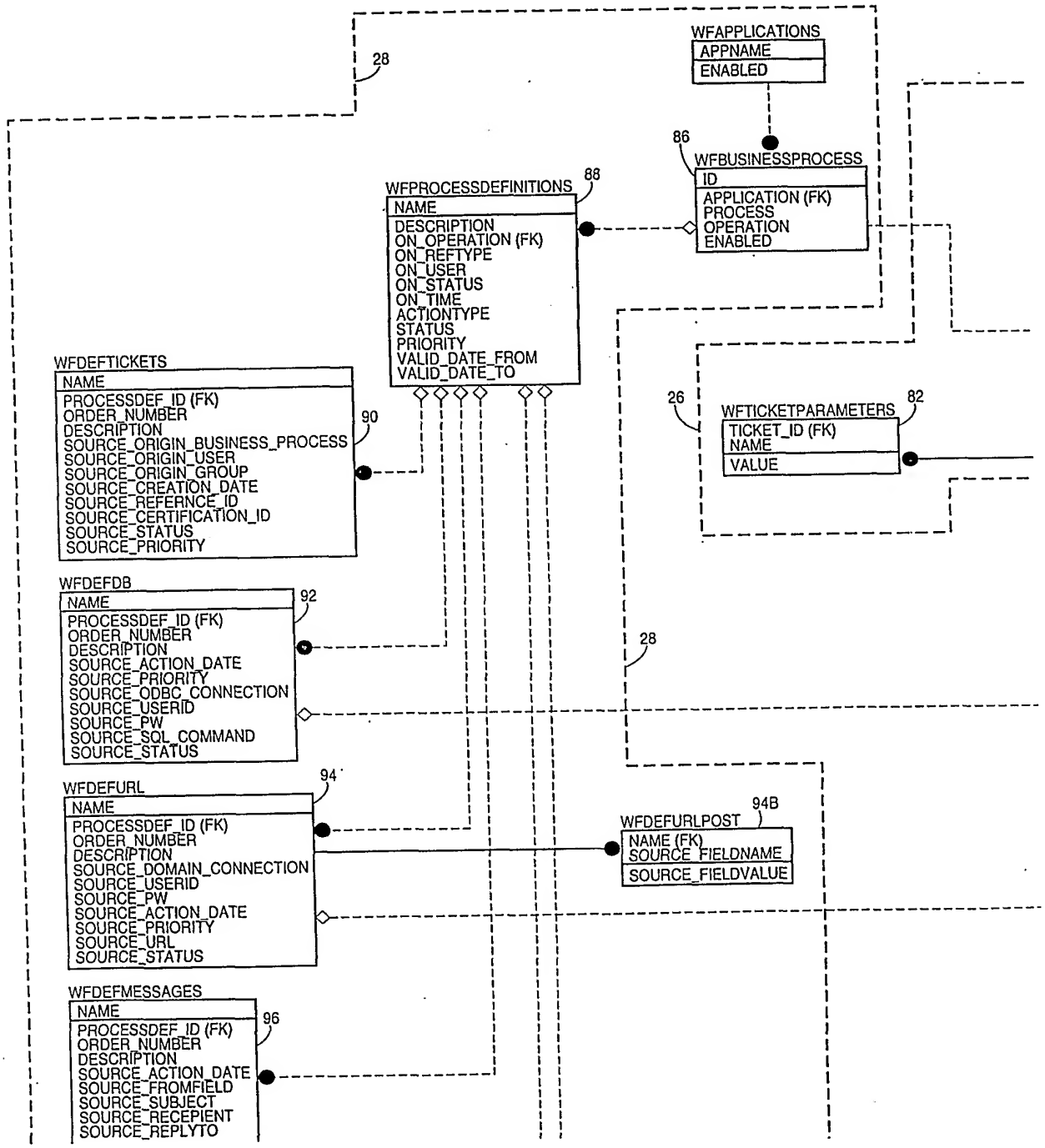

28
WFAPPLICATIONS
APPNAME
ENABLED
86
WFBUSINESSPROCESS
ID
APPLICATION (FK)
PROCESS
OPERATION
ENABLED
88
WFPROCESSDEFINITIONS
NAME
DESCRIPTION
ON_OPERATION (FK)
ON_REFTYPE
ON_USER
ON_STATUS
ON_TIME
ACTIONTYPE
STATUS
PRIORITY
VALID_DATE_FROM
VALID_DATE_TO
WFDEFTICKETS
NAME
PROCESSDEF_ID (FK)
ORDER_NUMBER
DESCRIPTION
SOURCE_ORIGIN_BUSINESS_PROCESS
SOURCE_ORIGIN_USER
SOURCE_ORIGIN_GROUP
SOURCE_CREATION_DATE
SOURCE_REFERNCE_ID
SOURCE_CERTIFICATION_ID
SOURCE_STATUS
SOURCE_PRIORITY
90
26
WFTICKETPARAMETERS
TICKET_ID (FK)
NAME
VALUE
82
WFDEFDB
NAME
PROCESSDEF_ID (FK)
ORDER_NUMBER
DESCRIPTION
SOURCE_ACTION_DATE
SOURCE_PRIORITY
SOURCE_ODBC_CONNECTION
SOURCE_USERID
SOURCE_PW
SOURCE_SQL_COMMAND
SOURCE_STATUS
92
28
WFDEFURL
NAME
PROCESSDEF_ID (FK)
ORDER_NUMBER
DESCRIPTION
SOURCE_DOMAIN_CONNECTION
SOURCE_USERID
SOURCE_PW
SOURCE_ACTION_DATE
SOURCE_PRIORITY
SOURCE_URL
SOURCE_STATUS
94
94B
WFDEFURLPOST
NAME (FK)
SOURCE_FIELDNAME
SOURCE_FIELDVALUE
WFDEFMESSAGES
NAME
PROCESSDEF_ID (FK)
ORDER_NUMBER
DESCRIPTION
SOURCE_ACTION_DATE
SOURCE_FROMFIELD
SOURCE_SUBJECT
SOURCE_RECEPIENT
SOURCE_REPLYTO
96

Figure 3a

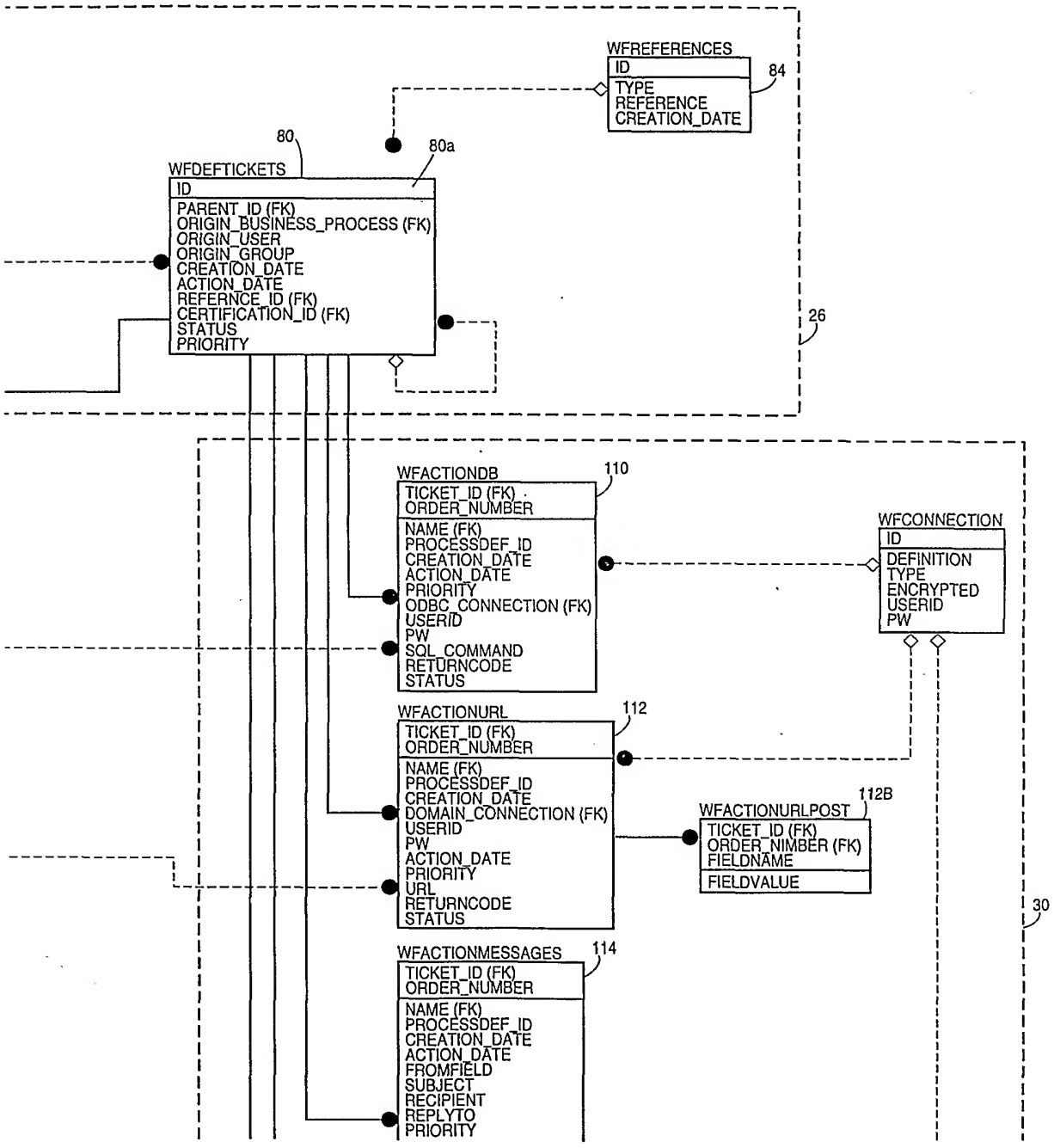

WFREFERENCES
ID
TYPE
REFERENCE
CREATION_DATE
84
80
80a
WFDEFTICKETS
ID
PARENT_ID (FK)
ORIGIN_BUSINESS_PROCESS (FK)
ORIGIN_USER
ORIGIN_GROUP
CREATION_DATE
ACTION_DATE
REFERNCE_ID (FK)
CERTIFICATION_ID (FK)
STATUS
PRIORITY
26
WFACTIONDB
110
TICKET_ID (FK)
ORDER_NUMBER
NAME (FK)
PROCESSDEF_ID
CREATION_DATE
ACTION_DATE
PRIORITY
ODBC_CONNECTION (FK)
USERID
PW
SQL_COMMAND
RETURNCODE
STATUS
WFCONNECTION
ID
DEFINITION
TYPE
ENCRYPTED
USERID
PW
WFACTIONURL
112
TICKET_ID (FK)
ORDER_NUMBER
NAME (FK)
PROCESSDEF_ID
CREATION_DATE
DOMAIN_CONNECTION (FK)
USERID
PW
ACTION_DATE
PRIORITY
URL
RETURNCODE
STATUS
112B
WFACTIONURLPOST
TICKET_ID (FK)
ORDER_NIMBER (FK)
FIELDNAME
FIELDVALUE
30
WFACTIONMESSAGES
114
TICKET_ID (FK)
ORDER_NUMBER
NAME (FK)
PROCESSDEF_ID
CREATION_DATE
ACTION_DATE
FROMFIELD
SUBJECT
RECIPIENT
REPLYTO
PRIORITY

Figure 3b

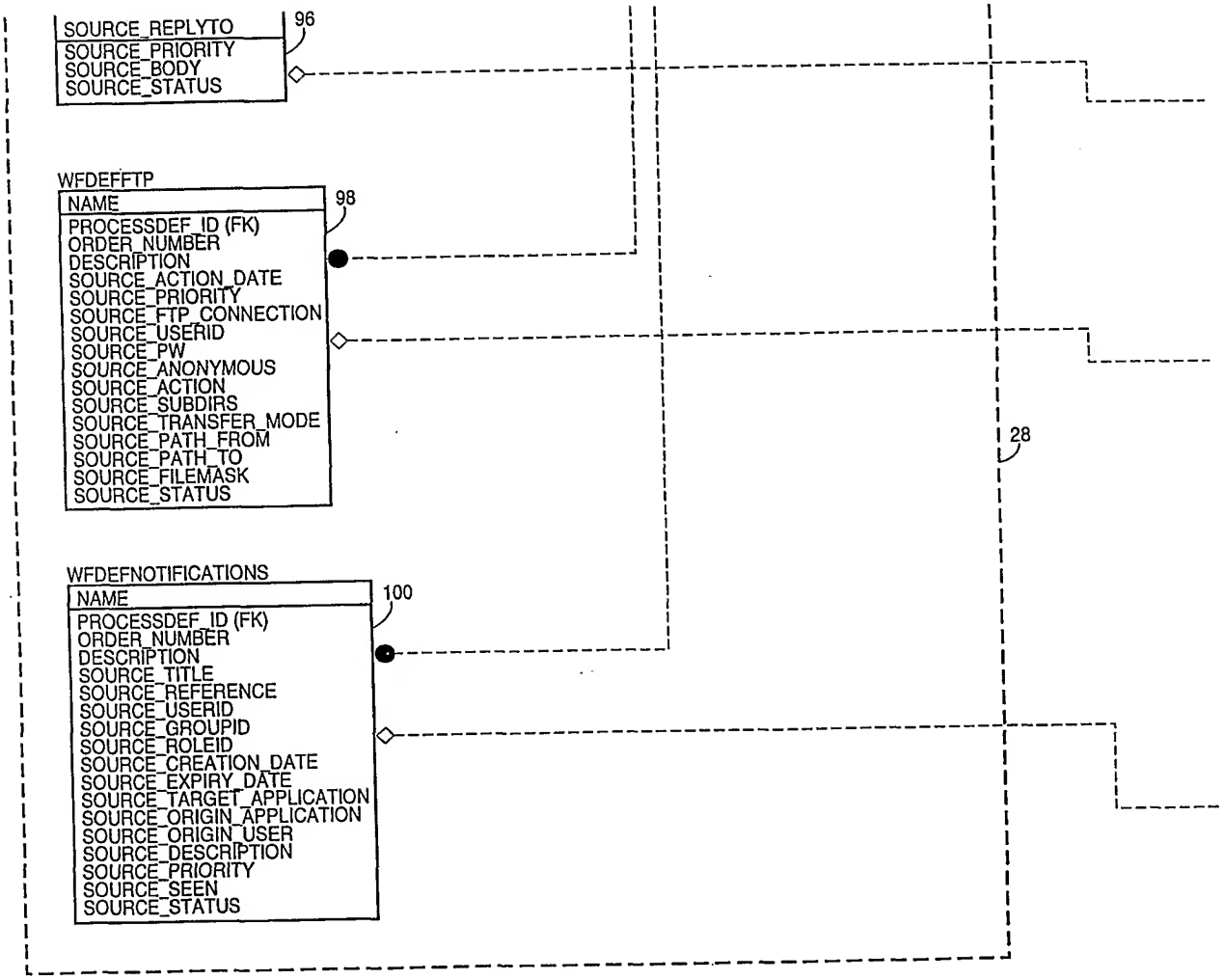

SOURCE_REPLYTO
SOURCE_PRIORITY
SOURCE_BODY
SOURCE_STATUS
96
WFDEFFTP
NAME
PROCESSDEF_ID (FK)
ORDER_NUMBER
DESCRIPTION
SOURCE_ACTION_DATE
SOURCE_PRIORITY
SOURCE_FTP_CONNECTION
SOURCE_USERID
SOURCE_PW
SOURCE_ANONYMOUS
SOURCE_ACTION
SOURCE_SUBDIRS
SOURCE_TRANSFER_MODE
SOURCE_PATH_FROM
SOURCE_PATH_TO
SOURCE_FILEMASK
SOURCE_STATUS
98
28
WFDEFNOTIFICATIONS
NAME
PROCESSDEF_ID (FK)
ORDER_NUMBER
DESCRIPTION
SOURCE_TITLE
SOURCE_REFERENCE
SOURCE_USERID
SOURCE_GROUPID
SOURCE_ROLEID
SOURCE_CREATION_DATE
SOURCE_EXPIRY_DATE
SOURCE_TARGET_APPLICATION
SOURCE_ORIGIN_APPLICATION
SOURCE_ORIGIN_USER
SOURCE_DESCRIPTION
SOURCE_PRIORITY
SOURCE_SEEN
SOURCE_STATUS
100

Figure 3c

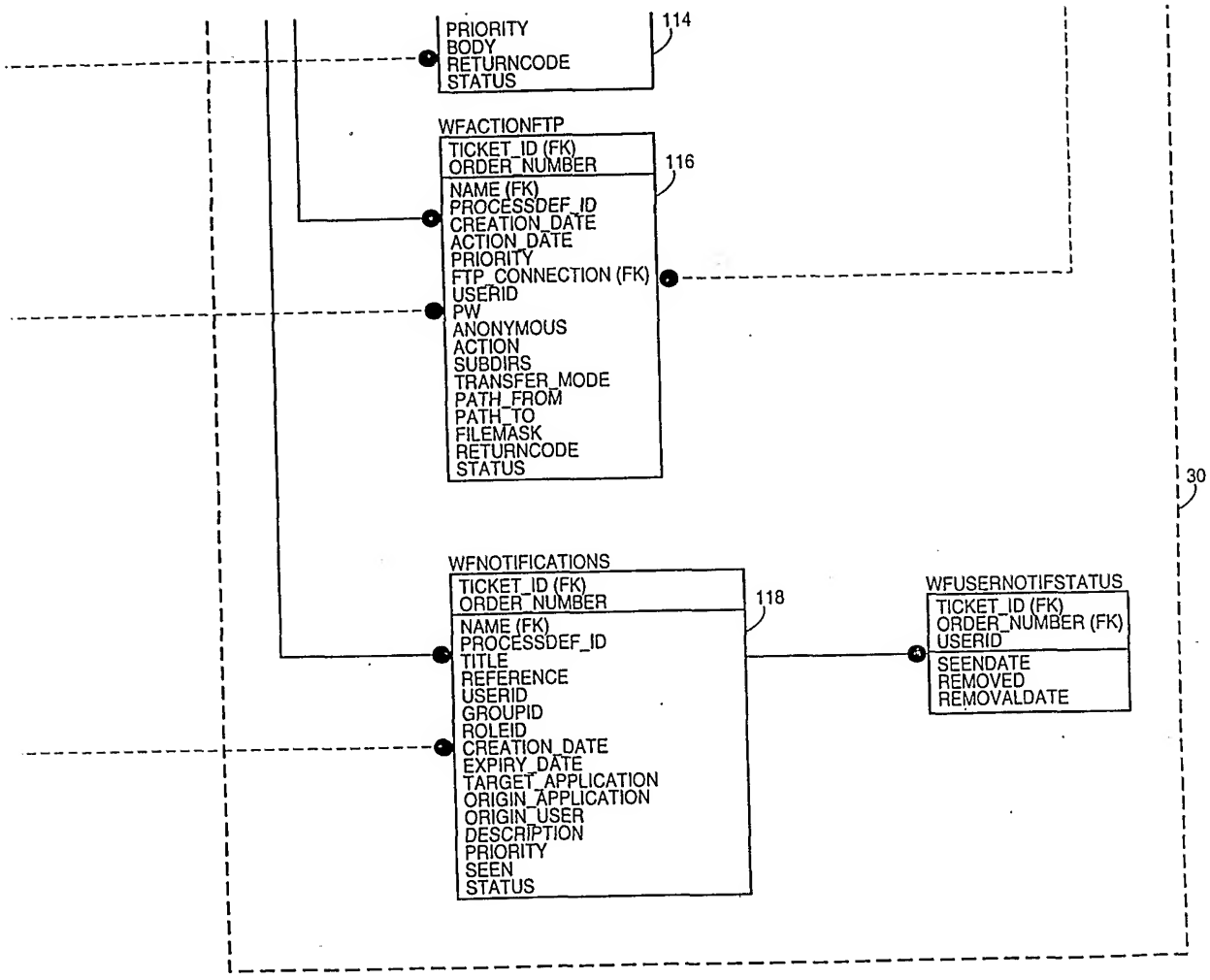

PRIORITY
BODY
RETURNCODE
STATUS
114
WFACTIONFTP
TICKET_ID (FK)
ORDER_NUMBER
NAME (FK)
PROCESSDEF_ID
CREATION_DATE
ACTION_DATE
PRIORITY
FTP_CONNECTION (FK)
USERID
PW
ANONYMOUS
ACTION
SUBDIRS
TRANSFER_MODE
PATH_FROM
PATH_TO
FILEMASK
RETURNCODE
STATUS
116
30
WFNOTIFICATIONS
TICKET_ID (FK)
ORDER_NUMBER
NAME (FK)
PROCESSDEF_ID
TITLE
REFERENCE
USERID
GROUPID
ROLEID
CREATION_DATE
EXPIRY_DATE
TARGET_APPLICATION
ORIGIN_APPLICATION
ORIGIN_USER
DESCRIPTION
PRIORITY
SEEN
STATUS
118
WFUSERNOTIFSTATUS
TICKET_ID (FK)
ORDER_NUMBER (FK)
USERID
SEENDATE
REMOVED
REMOVALDATE

Figure 3d

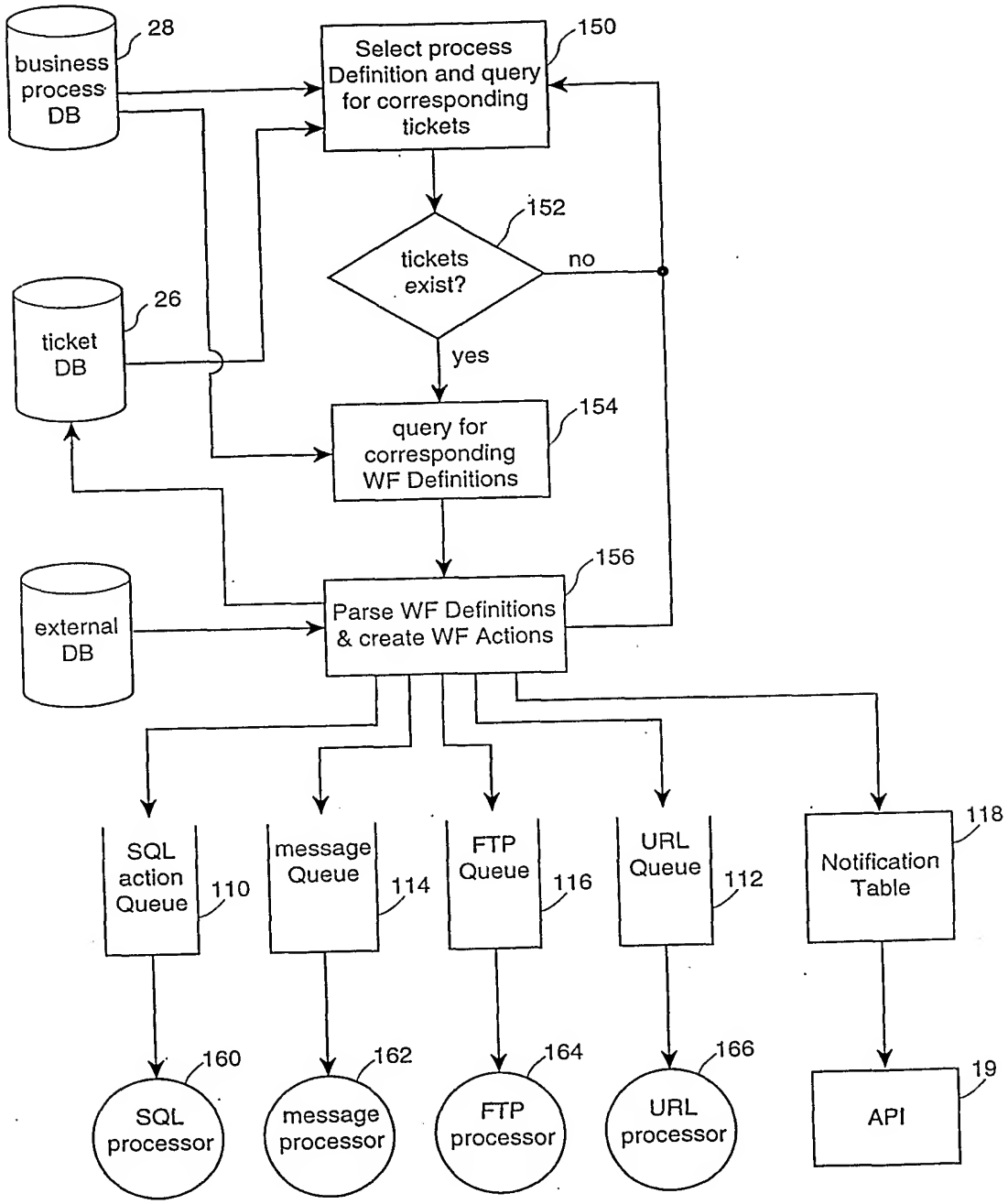

business process DB
28
Select process Definition and query for corresponding tickets
150
tickets exist?
152
no
yes
ticket DB
26
query for corresponding WF Definitions
154
external DB
Parse WF Definitions & create WF Actions
156
SQL action Queue
110
message Queue
114
FTP Queue
116
URL Queue
112
Notification Table
118
SQL processor
160
message processor
162
FTP processor
164
URL processor
166
API
19

Figure 4